**TABLE OF CONTENTS**

**ABSTRACT**

1. **SYSTEM STUDY**
   1. Feasibility Study
      1. Economical Feasibility
      2. Technical Feasibility
      3. Operational Feasibility
      4. Legal Feasibility

* 1. System Requirements
     1. Hardware Requirements
     2. Software Requirements

1. **SYSTEM ANALYSIS**
   1. Introduction
   2. Project Description
   3. Modules
      1. Social Network model
      2. Epidemic Model
      3. Feedback Model
   4. Existing System
      1. Drawbacks of Existing System
   5. Proposed system
      1. Advantage of Proposed System
2. **SYSTEM DESIGN**
   1. Overall Diagram
   2. Architecture Diagram
   3. Case Diagram
   4. Class Diagram
   5. Architecture Diagram
   6. Source Code
3. **TESTING**
   1. Unit Testing
   2. Integration Testing
   3. Validation Testing
   4. Acceptance Testing
4. **SYSTEM TESTING & IMPLEMENTATION**
   1. Introduction
   2. Strategic approach to Software testing
   3. Unit Testing
   4. Software Environment
5. **Process**
   1. Sending Data to the Cloud:
   2. Data Preprocessing:
   3. Feature Selection:
   4. Training the Random Forest Algorithm:
   5. Testing the Model:
   6. Deploying the Model:
   7. Predicting Cancer:
6. **CONCLUSIONS**
   1. Conclusion
   2. Future Enhancements
7. **REFERENCES**

**ABSTRACT**

The use of emerging technologies such as Internet of Things (IoT), Cloud Computing, Deep Learning, Machine Learning, and Artificial Intelligence (AI) is becoming increasingly important in various fields, including healthcare systems, transportation systems, agriculture, and smart cities. In this research work, the focus is on the application of IoT and cloud computing in the field of healthcare. Specifically, the research aims to design a cancer prediction system using IoT that collects patient-related data from sensors deployed in the human body. The collected data, including blood results, are encrypted and stored in the cloud for quick reference and analysis by doctors or healthcare nurses.

The cancer prediction system is designed to test whether the collected blood results are normal or abnormal. This system uses encryption algorithms, such as AES, to ensure the security and confidentiality of the patient's data. The main advantage of using cloud computing in healthcare is the ability to store and access medical data from anywhere, thus overcoming the limitations of traditional medical treatment.

The proposed framework enhances the performance of the existing healthcare industry across the globe by providing a more efficient and effective way of handling healthcare data. The focus is on handling cancer patient data effectively, even when they are away from their hometown, as the required treatment details are stored in the cloud. The use of virtual machines (VMs) reduces the task completion time significantly from 400 to 160. The simulation results can be modeled and reproduced using CloudSim, which provides an adaptable simulation structure.

In summary, the research work focuses on the design and implementation of a cancer prediction system using IoT and cloud computing. The system is designed to collect patient-related data, including blood results, encrypt them for security, and store them in the cloud for quick reference and analysis. The proposed framework provides a more efficient and effective way of handling healthcare data and enhances the performance of the existing healthcare industry

**CHAPTER -1**

**SYSTEM STUDY**

* 1. **Feasibility Study**

A feasibility study is an important step in determining the viability of a project, and it involves evaluating the technical, economic, and operational aspects of the project. In the case of a cancer prediction system using IoT and cloud computing. This is to ensure that the proposed system is not a burden. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* OPERATIONAL FEASIBILITY
* LEGAL FEASIBILITY
  + 1. **Economical Feasibility**

The economic feasibility of the project is an essential factor to consider, as it involves determining the financial viability of the project. The project may require significant investments in hardware, software, and cloud infrastructure. However, the benefits of the system in terms of improving healthcare and saving lives may outweigh the costs. The economic feasibility of the project can be justified by the potential benefits of the system in terms of improving healthcare outcomes, reducing healthcare costs, and providing a competitive advantage to healthcare providers. However, a detailed cost-benefit analysis should be conducted to ensure that the costs of implementing the system do not outweigh the benefits.

* + 1. **Technical Feasibility**

The project's technical feasibility involves determining if the technology required to implement the system is readily available and if it is capable of meeting the project's requirements. The project's technical feasibility is high, as the required technologies such as IoT sensors, cloud infrastructure, and machine learning algorithms are readily available.

* + 1. **Operational Feasibility**

The operational feasibility of the project involves determining if the system can be integrated into the existing healthcare infrastructure and if it can be operated and maintained effectively. The project may require training for medical staff to use the system effectively.

* + 1. **Legal Feasibility**

The legal feasibility of the project involves determining if the system complies with relevant laws and regulations, such as data protection laws. Appropriate measures such as encryption algorithms such as AES can be used to ensure the confidentiality and integrity of the patient's data.

* 1. **System Requirements**
     1. **Hardware Requirements**

# Processor - I3 & above

RAM - 4 GB

Hard Disk - 500 GB

* + 1. **Software Requirements**

Operating System – Windows 8/10/11

Cloud service provider- AWS(Amazon Web Services)

**CHAPTER-2**

**SYSTEM ANALYSIS**

* 1. **Introduction**

The Internet of Things (IoT) has emerged as a game-changing technology in all fields, transforming the way we live, work, and interact with the world around us. IoT is essentially a network of interconnected devices, machines, animals, people, and objects that are given unique identifiers and the ability to move data over a network without human intervention.

In the healthcare industry, IoT has revolutionized the way healthcare is delivered by enabling healthcare providers to remotely monitor and manage patient health. The deployment of various sensors in the human body can predict the heartbeats, blood pressure, and variations in the functioning of human body parts, enabling doctors and healthcare professionals to diagnose and treat patients more effectively and efficiently.The IoT has also enabled the development of smart cities, transportation systems, and e-healthcare systems, among others, by connecting various devices and machines to the internet and enabling them to communicate with each other without human intervention. This has led to the creation of a vast network of billions of connected devices and machines that will soon join human clients in the digital world.

The IoT cloud service has played a significant role in facilitating communication among the sensors in the IoT network, making it more efficient and effective. The cloud service enables the storage, processing, and analysis of vast amounts of data generated by IoT devices, allowing healthcare providers to access patient data remotely and in real-time, thereby improving the quality of healthcare services.

In conclusion, IoT is a revolutionary technology that is transforming the way we live and work, and its potential is immense. With the growing generation of data, the IoT cloud service will become more critical in facilitating communication and processing of data, and its role in healthcare will continue to expand, leading to improved patient outcomes and better healthcare services.

* 1. **Project Description**

In this research work, we aim to design a cancer prediction system using Internet of Things (IoT) and cloud computing technologies for the healthcare industry. With the increasing importance of emerging technologies like IoT, cloud computing, and Artificial Intelligence (AI), their application in various fields is becoming more and more crucial. The healthcare system is one such field where these technologies can be used to enhance the performance of the existing system.

The proposed cancer prediction system will collect patient-related data from sensors deployed in the human body using IoT. The data collected will include vital information such as body temperature, blood pressure, and other factors that can predict the onset of cancer. This information will be processed using cloud computing technologies to generate insights and predictions that can help doctors and healthcare professionals make informed decisions.

The collected data will be encrypted and stored in the cloud to ensure the privacy and security of the patient's sensitive medical information. This will also enable doctors and healthcare nurses to access the patient's medical data quickly, even if they are away from their physical location.

The system will be developed using various technologies, including Deep Learning and Machine Learning, to enhance the performance and accuracy of the cancer prediction system. The proposed system's success will not only help doctors and healthcare professionals make informed decisions, but it will also help patients receive timely treatment for cancer, ultimately saving lives.

Overall, the proposed cancer prediction system is a step forward in enhancing the healthcare industry's performance and providing better care for patients worldwide. With the integration of IoT and cloud computing technologies, we can overcome the traditional medical treatment limitations and improve healthcare computations and processing.

* 1. **Modules**
     1. **Data collection model**

With the proliferation of social networks and their ever increasing use, viruses have become much more prevalent. In this module the user login to the application and use search engine to search any content of data in the application to get the required data with respective to the keywords entered in the search engine.

* + 1. **IOT Model**

The user click the unofficial links and get access the data along with virus which get affected along with the retrieval of data then application. In a static network, weakly connected heterogeneous communities can have significantly different infection levels.

* + 1. **Cancer prediction Model**

This module would analyze the collected data to predict the likelihood of cancer based on the variations in the blood cell count and other relevant parameters**.**

* + 1. **Cloud computing Model**

This module would involve the use of cloud computing technologies to store the collected data securely in the cloud. The data would be encrypted to ensure the privacy and confidentiality of the patients.

* + 1. **Data analysis Model**

This module would enable doctors and healthcare nurses to analyze the data stored in the cloud and visualize it in a meaningful way to aid in diagnosis and treatment.

* + 1. **Data visualization Model**

This module would enable doctors and healthcare nurses to analyze the data stored in the cloud and visualize it in a meaningful way to aid in diagnosis and treatment.

* + 1. **Machine learning & Deep learning Model**

This module would involve the use of machine learning and deep learning algorithms to train the system on large datasets to improve the accuracy of cancer prediction and treatment recommendations.

* + 1. **Security Model**

This module would ensure the security and privacy of the patient data by implementing authentication, authorization, and encryption mechanisms. It would also involve monitoring the system for any potential security breaches and taking appropriate action to prevent them.

* 1. **Existing System**

The existing healthcare system typically relies on traditional methods of patient data collection and analysis, which are often manual and time-consuming. The healthcare professionals collect patient data manually and store it in paper files or electronic health records (EHRs) in a central database. However, these traditional methods are prone to errors and can lead to delays in diagnosis and treatment. Moreover, the traditional healthcare system lacks real-time monitoring and analysis of patient data, which could result in missed diagnoses and delayed treatment.

* + 1. .**Drawbacks of Existing System**
* Limited accuracy
* Delayed diagnosis
* Limited accessibility
  1. **Proposed system**

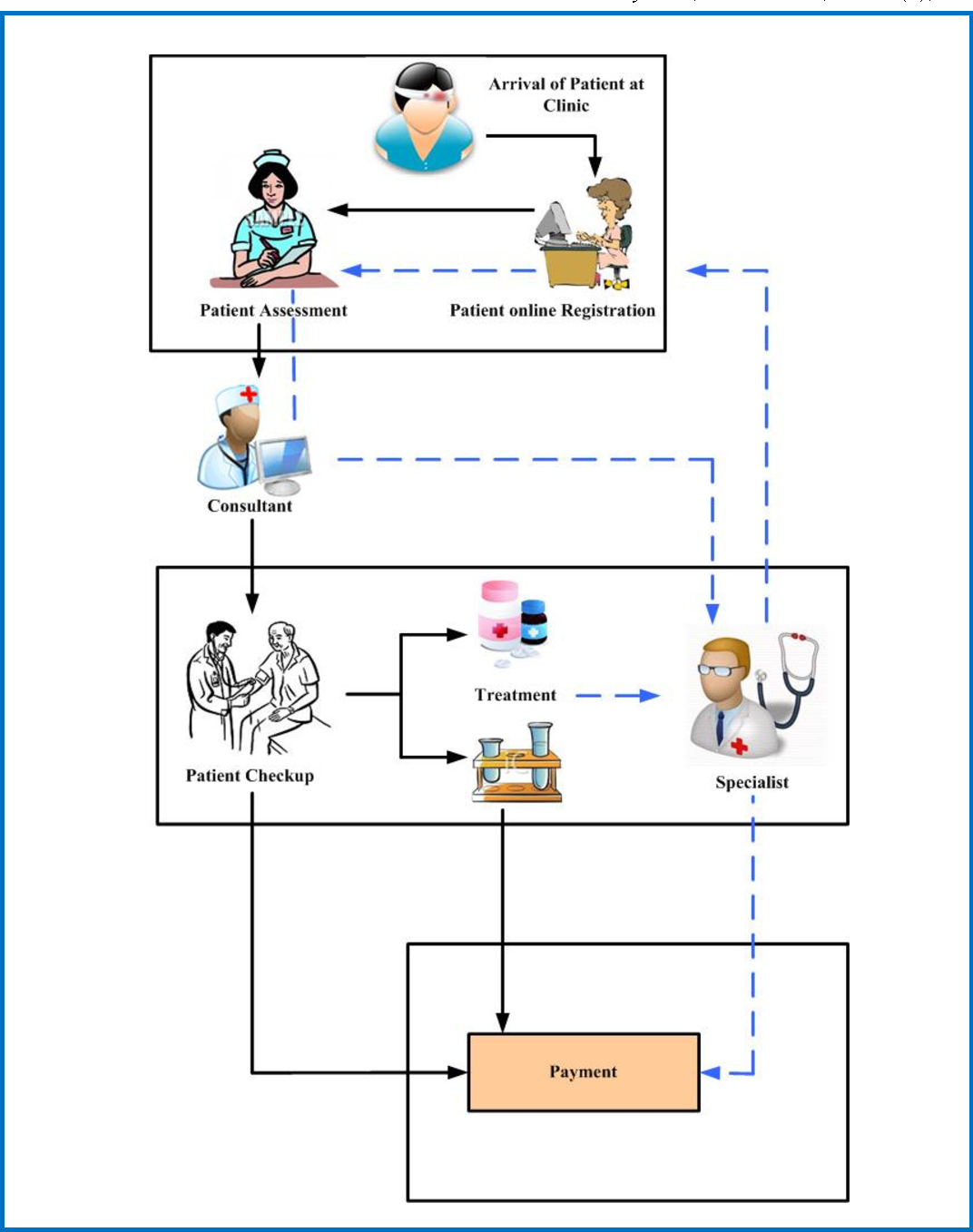
The proposed system is a cancer prediction system that utilizes Internet of Things (IoT) and cloud computing to overcome the limitations of the traditional cancer prediction system. The system is designed to collect patient-related data, including blood results, from sensors deployed in the human body through IoT. The collected data is then encrypted and stored in the cloud for quick and secure access by doctors or healthcare nurses.

* + 1. **Advantage of Proposed System**
* Real-time monitoring
* Easily accessible
* Quickly and securely

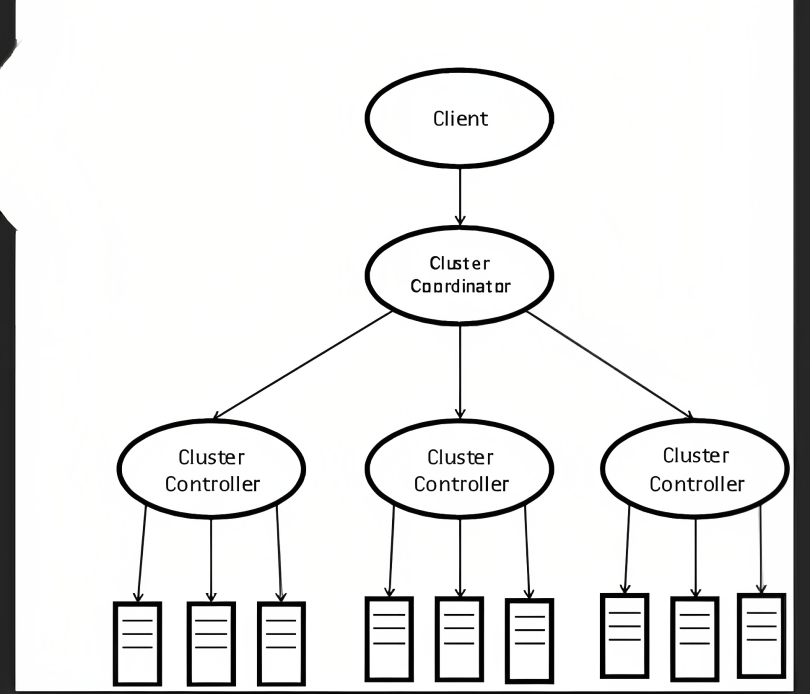
**CHAPTER - 3**

**SYSTEM DESIGN**

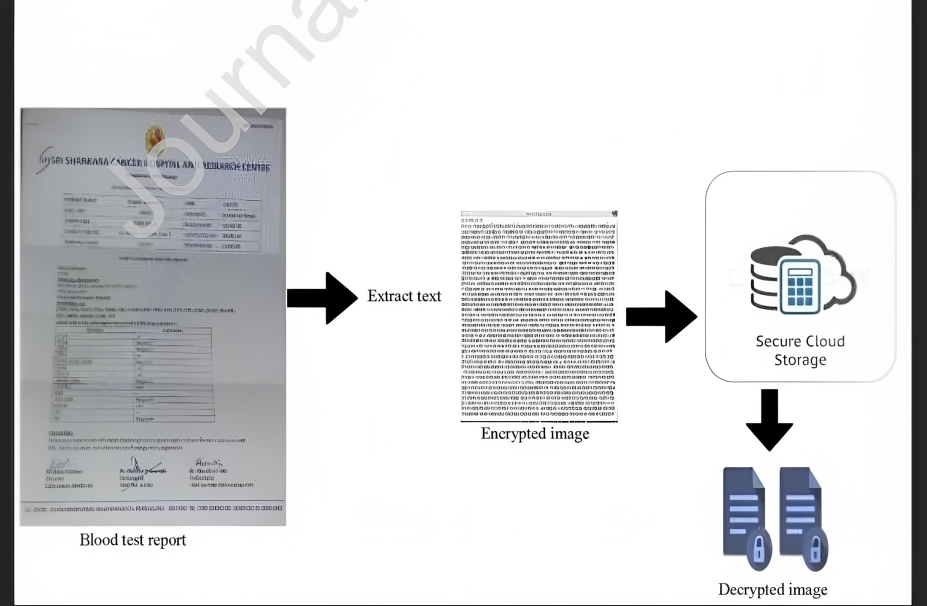
**LEVEL-0 (rural system)**

****

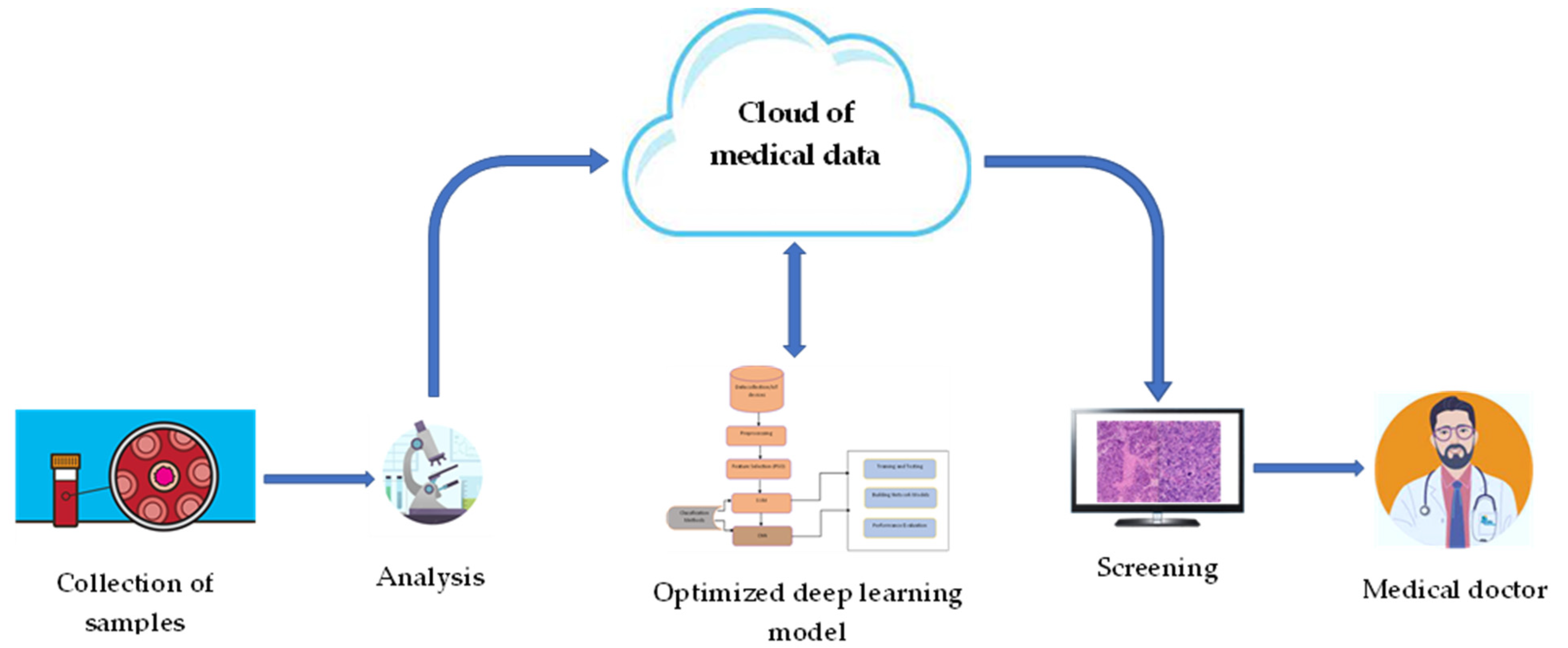
**LEVEL-1**

****

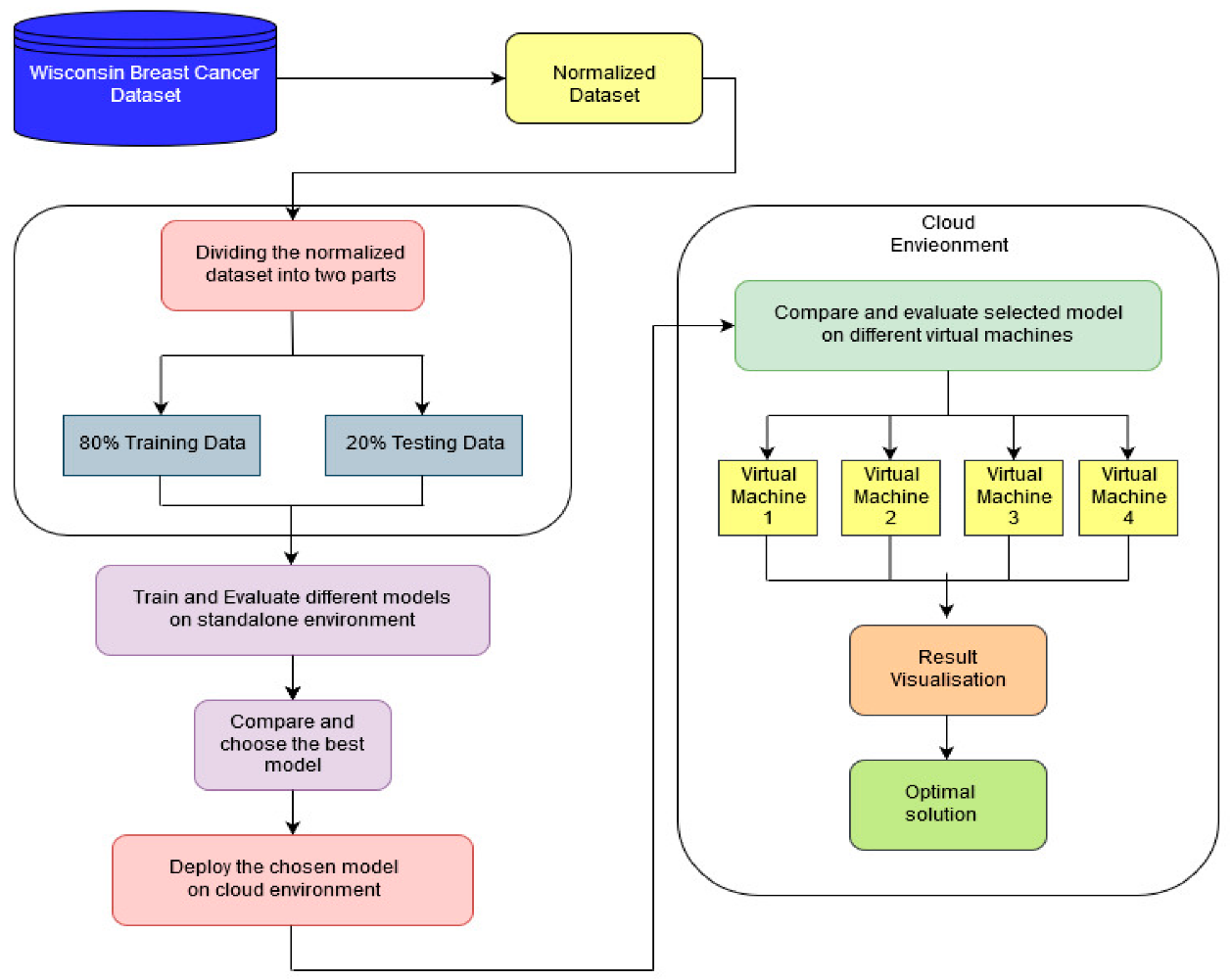
**LEVEL-2**

****

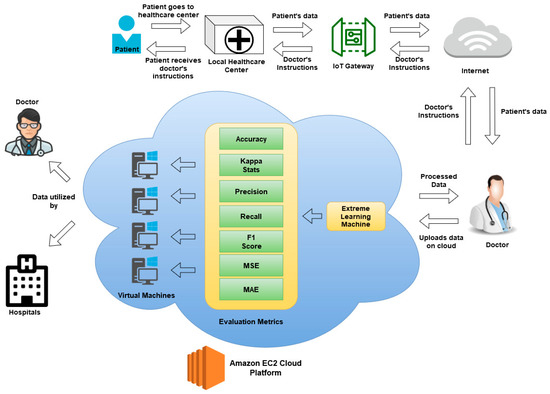
**Overall Diagram**



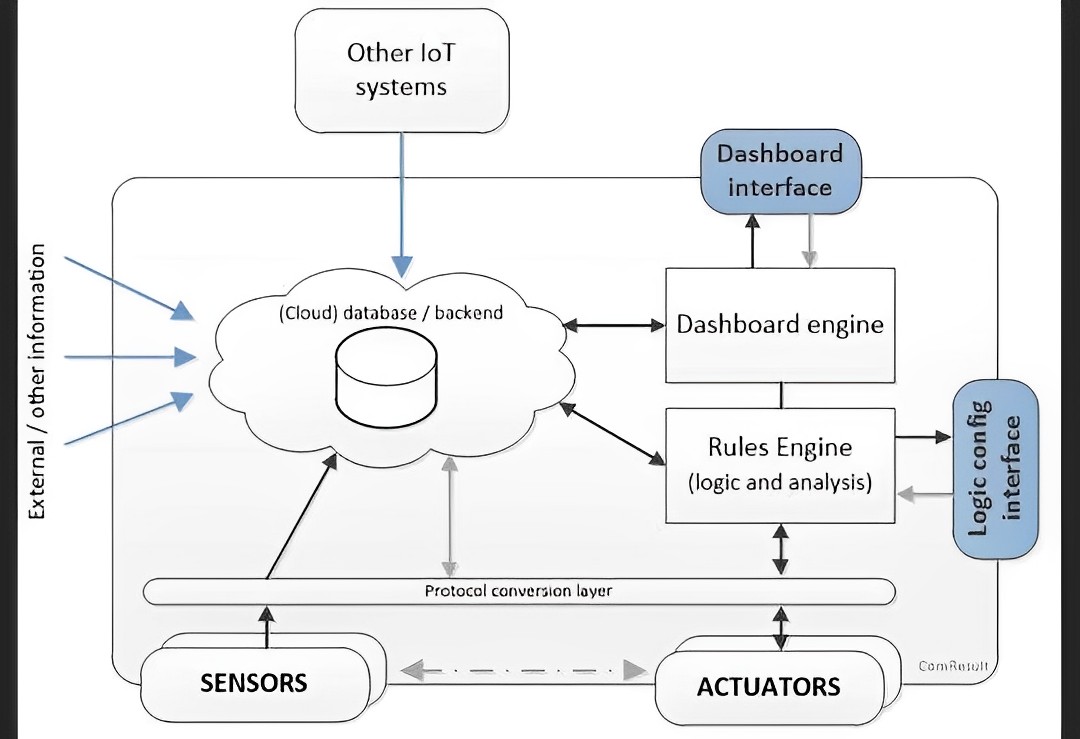
* 1. **Case Diagram**

****

* 1. **Class Diagram**

****

* 1. **Architecture Diagram**

****

* 1. **Source Code**

**CODING:**

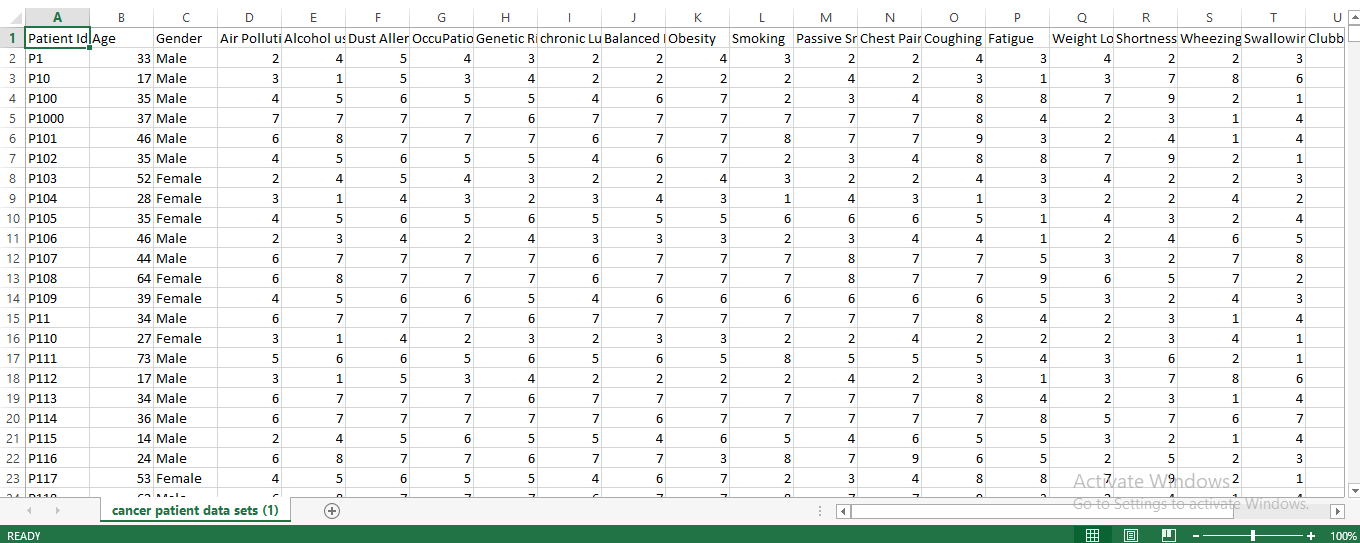
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

df = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

****

import pandas as pd

import numpy as np

import seaborn as sns

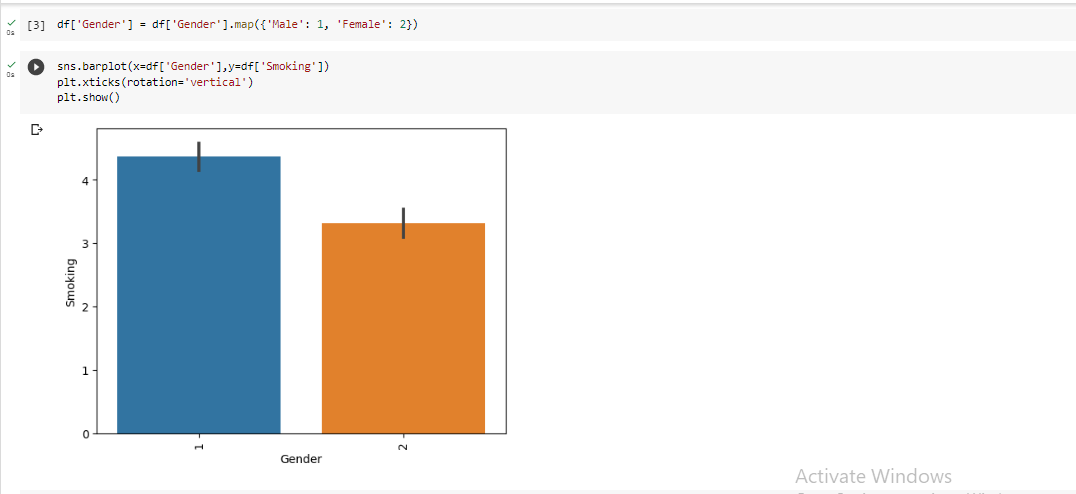
import matplotlib.pyplot as plt

df = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

df.isnull().sum()

df = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

df['Gender'] = df['Gender'].map({'Male': 1, 'Female': 2})



import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

df = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

df.isnull().sum()

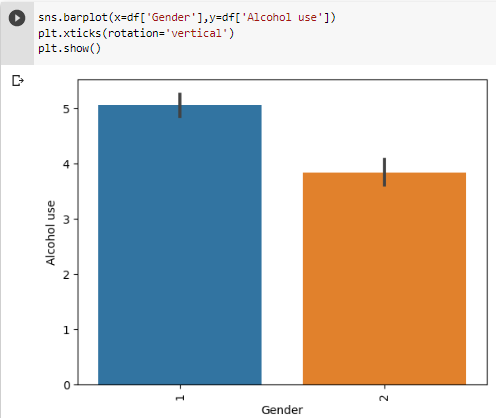
df = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

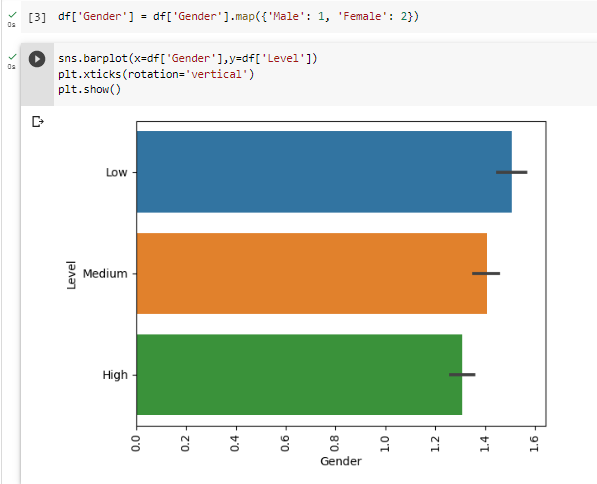
df['Gender'] = df['Gender'].map({'Male': 1, 'Female': 2})

sns.barplot(x=df['Gender'],y=df['Alcohol use'])

plt.xticks(rotation='vertical')

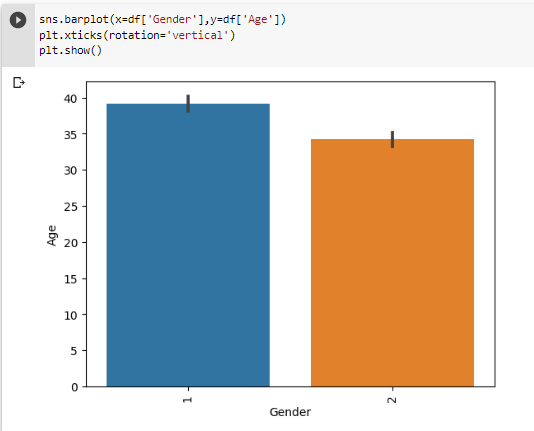
plt.show()



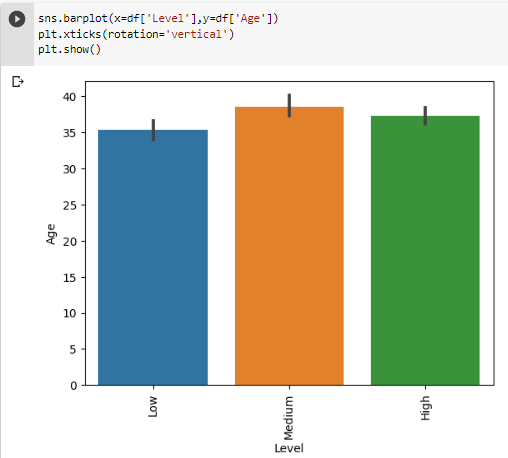


sns.barplot(x='Gender', y='Age', data=df)

plt.show()

****

sns.displot(df['Level'].dropna())

****

df.drop('Patient Id', axis=1, inplace=True)

import pandas as pd

# assume this is your input data

data = pd.DataFrame({

    'Age': [30, 40, 20, 25],

    'OccuPational Hazards': ['Engineer', 'Teacher', 'Doctor', 'Engineer'],

    'chronic Lung Disease': ['1', '0', '0', '1']

})

# define the categorical features to be one-hot encoded

categorical\_features = ['Age', 'OccuPational Hazards', 'chronic Lung Disease']

# use pandas get\_dummies method to one-hot encode the categorical features

encoded\_data = pd.get\_dummies(data, columns=categorical\_features)

# print the encoded data

print(encoded\_data.head())

import pandas as pd

from sklearn.preprocessing import StandardScaler

# Define the data and the features

data = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

numerical\_features =  ['Age', 'Air Pollution', 'Alcohol use', 'Dust Allergy', 'Genetic Risk',

                      'Balanced Diet', 'Obesity', 'Smoking', 'Passive Smoker', 'Chest Pain',

                      'Coughing of Blood', 'Fatigue', 'Weight Loss', 'Shortness of Breath',

                      'Wheezing', 'Swallowing Difficulty', 'Clubbing of Finger Nails',

                      'Frequent Cold', 'Dry Cough', 'Snoring']

# Separate the input features and the target variable

X = data[numerical\_features]

y = data['Level']

# Scale the input features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

from sklearn.model\_selection import train\_test\_split

# Split data into features and target variable

X = data[categorical\_features]

y = data['Level']

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Define the categorical features and one-hot encode them

categorical\_features =  ['Age', 'OccuPational Hazards', 'chronic Lung Disease']

X\_train\_encoded = pd.get\_dummies(X\_train, columns=categorical\_features)

X\_test\_encoded = pd.get\_dummies(X\_test, columns=categorical\_features)

from sklearn.model\_selection import cross\_val\_score

from sklearn.linear\_model import LogisticRegression

from sklearn.datasets import load\_iris

# Load the iris dataset

iris = load\_iris()

# Create X and y variables

X = iris.data

y = iris.target

# Create a logistic regression model

model = LogisticRegression()

# Perform cross-validation with 5 folds

cv\_scores = cross\_val\_score(model, X, y, cv=5)

# Print the cross-validation scores for each fold

print("Cross-validation scores:", cv\_scores)

# Print the average cross-validation score

print("Average cross-validation score:", cv\_scores.mean())

The average accuracy of the model is 98.0% based on the cross-validation scores. Cross-validation is a technique used to evaluate the performance of a machine learning model. In this case, the model was evaluated using k-fold cross-validation with k=5 (the data was split into 5 folds). The cross-validation scores represent the accuracy of the model on each fold of the data, and the average score is the average of those scores. An average score of 0.98 means that the model is performing very well on the data, with an accuracy of 98%.

# Import necessary libraries

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score

# Load the breast cancer dataset

cancer\_data = pd.read\_csv('/content/sample\_data/cancer patient data sets (1).csv')

# Remove the ID column

cancer\_data = cancer\_data.drop('Patient Id', axis=1)

cancer\_data['Gender'] = cancer\_data['Gender'].map({'Male': 1, 'Female': 2, 'Trans': 3})

# Map the target variable from 'M' (malignant) and 'B' (benign) to 1 and 0

cancer\_data['Level'] = cancer\_data['Level'].map({'Low': 1, 'Medium': 2, 'High': 3})

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(cancer\_data.drop('Level', axis=1),

                                                    cancer\_data['Level'],

                                                    test\_size=0.2,

                                                    random\_state=42)

# Scale the features using StandardScaler

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Train a logistic regression model

model = LogisticRegression()

model.fit(X\_train, y\_train)

# Make predictions on the testing set

y\_pred = model.predict(X\_test)

# Evaluate the model using various metrics

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

#print("Precision:", precision\_score(y\_test, y\_pred))

#print("Recall:", recall\_score(y\_test, y\_pred))

#print("F1 Score:", f1\_score(y\_test, y\_pred))

#print("ROC-AUC Score:", roc\_auc\_score(y\_test, y\_pred))

An accuracy of 1.0 means that the predictions made by a model are 100% correct. In other words, the model has made no errors in its predictions. This is the highest possible accuracy score and indicates that the model is performing perfectly on the given task or dataset. However, it's important to note that achieving 100% accuracy may not always be possible or desirable, and in some cases, it could be an indication of overfitting to the training data.

"# Import necessary libraries\n",

"import pandas as pd\n",

"import numpy as np\n",

"from sklearn.model\_selection import train\_test\_split\n",

"from sklearn.preprocessing import StandardScaler\n",

"from sklearn.linear\_model import LogisticRegression\n",

"from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score\n",

"\n",

"# Load the breast cancer dataset\n",

"import pandas as pd\n",

"\n",

"# Load the breast cancer dataset\n",

"cancer\_data = pd.read\_excel('/content/sample\_data/cancer-patient-data-sets-\_1\_.xlsx')\n",

"\n",

"\n",

"\n",

"# Remove the ID column \n",

"cancer\_data = cancer\_data.drop('Patient Id', axis=1)\n",

"\n",

"cancer\_data['Gender'] = cancer\_data['Gender'].map({'Male': 1, 'Female': 2, 'Trans': 3})\n",

"\n",

"\n",

"# Map the target variable from 'M' (malignant) and 'B' (benign) to 1 and 0\n",

"cancer\_data['Level'] = cancer\_data['Level'].map({'Low': 1, 'Medium': 2, 'High': 3})\n",

"\n",

"# Split the data into training and testing sets\n",

"X\_train, X\_test, y\_train, y\_test = train\_test\_split(cancer\_data.drop('Level', axis=1), \n",

" cancer\_data['Level'], \n",

" test\_size=0.2, \n",

" random\_state=42)\n",

"\n",

"# Scale the features using StandardScaler\n",

"scaler = StandardScaler()\n",

"X\_train = scaler.fit\_transform(X\_train)\n",

"X\_test = scaler.transform(X\_test)\n",

"\n",

"# Train a logistic regression model\n",

"model = LogisticRegression()\n",

"model.fit(X\_train, y\_train)\n",

"\n",

"# Make predictions on the testing set\n",

"y\_pred = model.predict(X\_test)\n",

"\n",

"# Evaluate the model using various metrics\n",

"print(\"Accuracy:\", accuracy\_score(y\_test, y\_pred))\n",

"\n",

"\n",

"# Predict the target variable for the test data\n",

"y\_pred = model.predict(X\_test)\n",

"\n",

"# Evaluate the model using various metrics\n",

"print(\"Accuracy:\", accuracy\_score(y\_test, y\_pred))\n",

"print(\"Precision:\", precision\_score(y\_test, y\_pred, average='weighted'))\n",

"\n"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "kxEIYO78dMcD",

"outputId": "d6cdc41c-09ae-40f6-d6f1-3d88e18b8008"

},

"execution\_count": 12,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Accuracy: 1.0\n",

"Accuracy: 1.0\n",

"Precision: 1.0\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score\n",

"\n",

"# Train the model on the training data\n",

"(X\_train, y\_train)\n",

"from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score\n",

"from sklearn.tree import DecisionTreeClassifier\n",

"\n",

"# Initialize a decision tree classifier\n",

"clf = DecisionTreeClassifier()\n",

"\n",

"\n",

"clf.fit(X\_train, y\_train)\n",

"# Predict the labels of the test data\n",

"y\_pred = clf.predict(X\_test)\n",

"\n",

"# Evaluate the model using various metrics\n",

"print(\"Accuracy:\", accuracy\_score(y\_test, y\_pred))\n",

"print(\"Precision:\", precision\_score(y\_test, y\_pred, average='micro'))\n",

"print(\"Recall:\", recall\_score(y\_test, y\_pred, average='micro'))\n",

"print(\"F1 Score:\", f1\_score(y\_test, y\_pred, average='micro'))\n",

"\n"

],

"metadata": {

"colab": {

"base\_uri": "https://localhost:8080/"

},

"id": "tomvH9Jug\_r3",

"outputId": "90edab16-477a-4a4c-9364-7d4b9b6217cb"

},

"execution\_count": 15,

"outputs": [

{

"output\_type": "stream",

"name": "stdout",

"text": [

"Accuracy: 1.0\n",

"Precision: 1.0\n",

"Recall: 1.0\n",

"F1 Score: 1.0\n"

]

}

]

},

{

"cell\_type": "code",

"source": [

"\n",

"import pandas as pd\n",

"from sklearn.ensemble import RandomForestClassifier\n",

"\n",

"from sklearn.preprocessing import LabelEncoder\n",

"\n",

"\n",

"# Drop any rows with missing values\n",

"cancer\_data.dropna(inplace=True)\n",

"\n",

"# Convert the 'Level' column to a category type\n",

"cancer\_data['Level'] = cancer\_data['Level'].astype('category')\n",

"\n",

"# Encode the 'Level' column to numerical values\n",

"label\_encoder = LabelEncoder()\n",

"cancer\_data['Level'] = label\_encoder.fit\_transform(cancer\_data['Level'])\n",

"\n",

"# Split the data into features (X) and target (y)\n",

"X = cancer\_data.drop('Level', axis=1)\n",

"y = cancer\_data['Level']\n",

"\n",

"# Train a Random Forest classifier on the data\n",

"clf = RandomForestClassifier(n\_estimators=100, random\_state=0)\n",

"clf.fit(X, y)\n",

"\n",

"# Train a Random Forest classifier on the data\n",

"clf = RandomForestClassifier(n\_estimators=100, random\_state=0)\n",

"clf.fit(X, y)\n",

"\n",

"# Get input from the user\n",

"age = int(input(\"Enter age: \"))\n",

"gender = input(\"Enter gender (M/F): \")\n",

"air\_pollution = int(input(\"Enter air pollution level (1-10): \"))\n",

"alcohol\_use = int(input(\"Enter alcohol use level (1-10): \"))\n",

"dust\_allergy = int(input(\"Enter dust allergy level (1-10): \"))\n",

"occupational\_hazards = int(input(\"Enter occupational hazards level (1-10): \"))\n",

"genetic\_risk = int(input(\"Enter genetic risk level (1-10): \"))\n",

"chronic\_lung\_disease = int(input(\"Enter chronic lung disease level (1-10): \"))\n",

"balanced\_diet = int(input(\"Enter balanced diet level (1-10): \"))\n",

"obesity = int(input(\"Enter obesity level (1-10): \"))\n",

"smoking = int(input(\"Enter smoking level (1-10): \"))\n",

"passive\_smoker = int(input(\"Enter passive smoker level (1-10): \"))\n",

"chest\_pain = int(input(\"Enter chest pain level (1-10): \"))\n",

"coughing\_of\_blood = int(input(\"Enter coughing of blood level (1-10): \"))\n",

"fatigue = int(input(\"Enter fatigue level (1-10): \"))\n",

"weight\_loss = int(input(\"Enter weight loss level (1-10): \"))\n",

"shortness\_of\_breath = int(input(\"Enter shortness of breath level (1-10): \"))\n",

"wheezing = int(input(\"Enter wheezing level (1-10): \"))\n",

"swallowing\_difficulty = int(input(\"Enter swallowing difficulty level (1-10): \"))\n",

"clubbing\_of\_finger\_nails = int(input(\"Enter clubbing of finger nails level (1-10): \"))\n",

"frequent\_cold = int(input(\"Enter frequent cold level (1-10): \"))\n",

"dry\_cough = int(input(\"Enter dry cough level (1-10): \"))\n",

"snoring = int(input(\"Enter snoring level (1-10): \"))\n",

"\n",

"\n",

"\n",

"\n",

"user\_input = pd.DataFrame({\n",

" 'Age': [age],\n",

" 'Air Pollution': [air\_pollution],\n",

" 'Alcohol use': [alcohol\_use],\n",

" 'Dust Allergy': [dust\_allergy],\n",

" 'Occupational Hazards': [occupational\_hazards],\n",

" 'Genetic Risk': [genetic\_risk],\n",

" 'Chronic Lung Disease': [chronic\_lung\_disease],\n",

" 'Balanced Diet': [balanced\_diet],\n",

" 'Obesity': [obesity],\n",

" 'Smoking': [smoking],\n",

" 'Passive Smoker': [passive\_smoker],\n",

" 'Chest Pain': [chest\_pain],\n",

" 'Coughing of Blood': [coughing\_of\_blood],\n",

" 'Fatigue': [fatigue],\n",

" 'Weight Loss': [weight\_loss],\n",

" 'Shortness of Breath': [shortness\_of\_breath],\n",

" 'Clubbing of Finger Nails': [clubbing\_of\_finger\_nails],\n",

" 'Frequent Cold': [frequent\_cold],\n",

" 'Dry Cough': [dry\_cough],\n",

" 'Snoring': [snoring],\n",

" 'Gender': [gender],\n",

" 'Swallowing Difficulty': [swallowing\_difficulty]\n",

"})\n",

"\n",

"# One-hot encode the \"Gender\" and \"Swallowing Difficulty\" columns\n",

"user\_input\_encoded = pd.get\_dummies(user\_input, prefix=['Gender', 'Swallowing Difficulty'], columns=['Gender', 'Swallowing Difficulty'])\n",

"\n",

"# Add the one-hot encoded columns to the user\_input DataFrame\n",

"user\_input = pd.concat([user\_input, user\_input\_encoded], axis=1)\n",

"\n",

"# Drop the original \"Gender\" and \"Swallowing Difficulty\" columns\n",

"user\_input.drop(['Gender', 'Swallowing Difficulty'], axis=1, inplace=True)\n",

"\n",

"# Predict whether the user has cancer or not\n",

"prediction = clf.predict(user\_input)\n",

"\n",

"# Print the prediction\n",

"if prediction == 1:\n",

" print(\"The user has cancer.\")\n",

"else:\n",

" print(\"The user does not have cancer.\")"

]

**CHAPTER – 4**

**TESTING**

Testing is an important process in software development that involves verifying and validating the functionality and performance of the software. In the context of the proposed system, testing would involve several stages, including:

**4.1 Unit Testing**

This involves testing individual components of the system to ensure that they work as expected. For example, testing the sensor data collection module to ensure that it collects and sends accurate data.

* 1. **Integration Testing**

This involves testing the system as a whole to ensure that all the components work together seamlessly. For example, testing the integration of the sensor data collection module with the cloud storage and analysis module.

* 1. **Validation Testing**

This involves testing the entire system to ensure that it meets the functional and non-functional requirements. For example, testing the cancer prediction module to ensure that it accurately predicts cancer based on the collected data.

* 1. **Acceptance Testing**

This involves testing the system with the end-users to ensure that it meets their requirements and expectations. For example, testing the user interface to ensure that it is user-friendly and easy to use.

**CHAPTER - 5**

**SYSTEM TESTING & IMPLEMENTATION**

* 1. **Introduction**

The Internet of Things (IoT) has emerged as a game-changing technology in all fields, transforming the way we live, work, and interact with the world around us. IoT is essentially a network of interconnected devices, machines, animals, people, and objects that are given unique identifiers and the ability to move data over a network without human intervention.

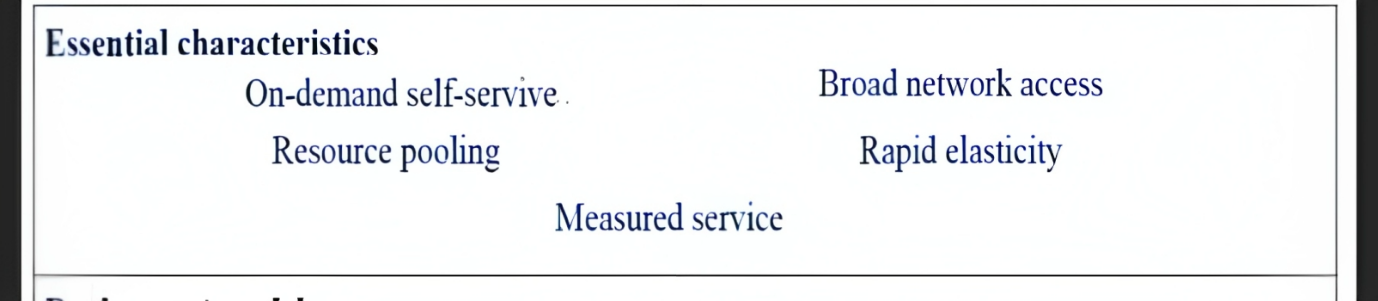
In the healthcare industry, IoT has revolutionized the way healthcare is delivered by enabling healthcare providers to remotely monitor and manage patient health. The deployment of various sensors in the human body can predict the heartbeats, blood pressure, and variations in the functioning of human body parts, enabling doctors and healthcare professionals to diagnose and treat patients more effectively and efficiently.The IoT has also enabled the development of smart cities, transportation systems, and e-healthcare systems, among others, by connecting various devices and machines to the internet and enabling them to communicate with each other without human intervention. This has led to the creation of a vast network of billions of connected devices and machines that will soon join human clients in the digital world.

The IoT cloud service has played a significant role in facilitating communication among the sensors in the IoT network, making it more efficient and effective. The cloud service enables the storage, processing, and analysis of vast amounts of data generated by IoT devices, allowing healthcare providers to access patient data remotely and in real-time, thereby improving the quality of healthcare services.

In conclusion, IoT is a revolutionary technology that is transforming the way we live and work, and its potential is immense. With the growing generation of data, the IoT cloud service will become more critical in facilitating communication and processing of data, and its role in healthcare will continue to expand, leading to improved patient outcomes and better healthcare services.

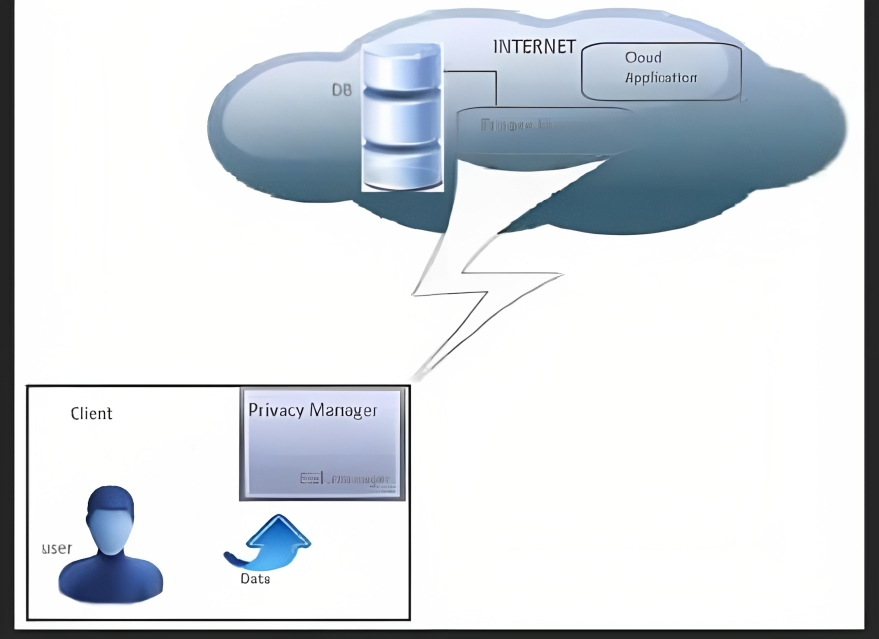
* 1. **CANCER PREDICTION SYSTEM USING IOT WITH CLOUD**

The Internet of Things (IoT) has indeed revolutionized the healthcare industry by enabling remote monitoring and management of patient health through the deployment of various sensors in the human body. This technology has made it possible to predict variations in the functioning of human body parts, making it easier for doctors and healthcare professionals to diagnose and treat patients more effectively and efficiently.

Moreover, the IoT has also facilitated the development of smart cities, transportation systems, and e-healthcare systems, among others, by connecting various devices and machines to the internet and enabling them to communicate with each other without human intervention. This has created a vast network of connected devices and machines that can communicate with each other, leading to increased efficiency and productivity in various industries. 

The IoT cloud service has played a critical role in enabling efficient communication among sensors in the IoT network, allowing for the storage, processing, and analysis of vast amounts of data generated by IoT devices. This service has made it possible for healthcare providers to access patient data remotely and in real-time, thereby improving the quality of healthcare services.

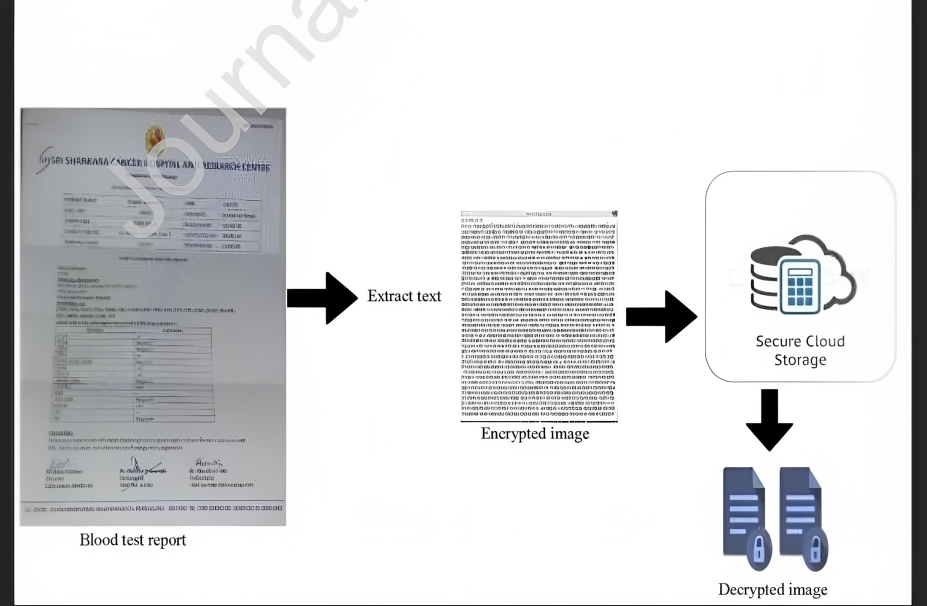
As the amount of data generated by IoT devices continues to grow, the IoT cloud service will become increasingly important in facilitating communication and processing of data. Its role in healthcare will continue to expand, leading to improved patient outcomes and better healthcare services.



In summary, the potential of IoT is immense, and it will continue to transform the way we live and work. The healthcare industry, in particular, will greatly benefit from this technology, leading to better patient outcomes and improved healthcare services.

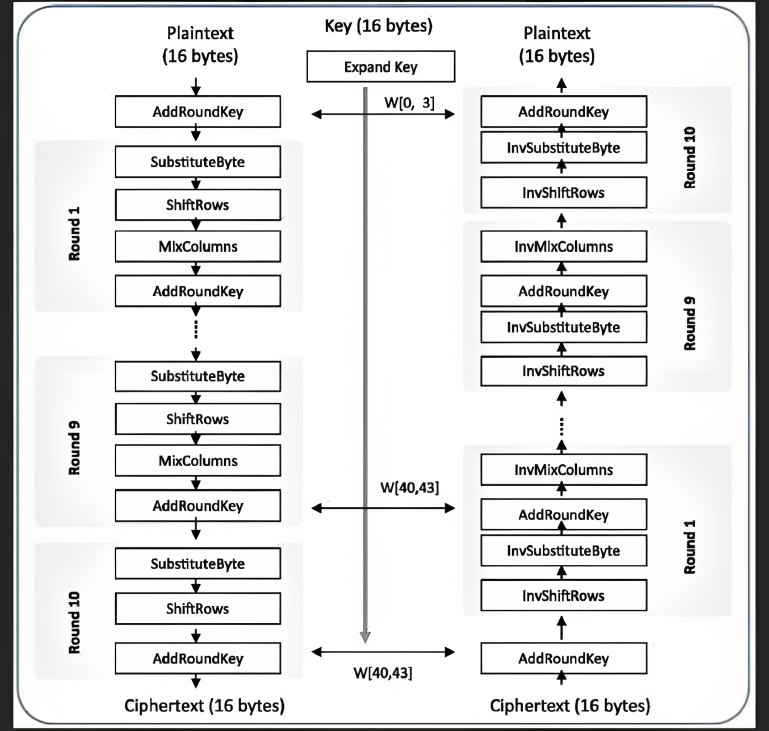
* 1. **Encryption**

The use of deep learning tools such as Alexnet, VGG 16, and GoogleNet for predicting the accuracy of different types of cancer is a crucial step towards improving the diagnosis and treatment of cancer patients. The proposed system utilizes the patient's blood test report or healthcare record as the input data, which is encrypted using the Advanced Encryption Standard (AES) algorithm to ensure the privacy and security of the patient's data. The use of AES algorithm for encryption is advantageous as it performs all its calculations on bytes and uses a variable number of rounds based on the length of the key, making it faster and stronger than Triple-DES.



Furthermore, storing the encrypted patient data in the cloud can significantly improve the performance of the e-healthcare system, as it enables patients to access their healthcare data from anywhere at any time without any delay or extensive processing. This feature is particularly beneficial for patients who need to move from their hometown for any reason, as they can continue their treatment without any interruption by accessing their stored e-healthcare data in the cloud.

Overall, the proposed system is a significant step towards improving the accuracy of cancer diagnosis and treatment by utilizing the power of deep learning tools and encryption algorithms like AES, and storing patient data in the cloud to provide seamless access to healthcare data.



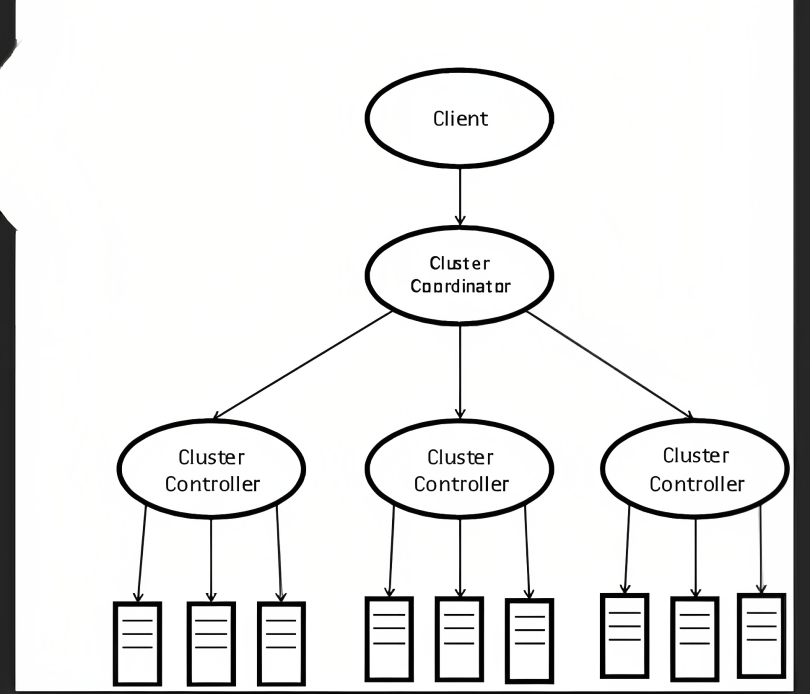
* 1. **Three tire architecture**

A three-tier architecture is a software architecture pattern that consists of three layers or tiers: a presentation layer, an application or business logic layer, and a data storage layer.

The presentation layer is the topmost layer and is responsible for displaying the user interface to the end-user. This layer interacts with the application layer to retrieve and manipulate data, but does not directly communicate with the data storage layer.

The application layer is the middle layer and contains the business logic or application logic of the system. This layer processes user requests from the presentation layer, interacts with the data storage layer to retrieve and manipulate data, and sends the processed data back to the presentation layer for display to the user.

The data storage layer is the bottommost layer and is responsible for storing and retrieving data. This layer interacts with the application layer to receive requests for data storage or retrieval and responds with the requested data.



A three-tier architecture provides several advantages such as improved scalability, modularity, and flexibility. It also enables easier maintenance and updates as each layer can be updated independently without affecting the other layers. Additionally, it can help to ensure better security by separating concerns and restricting access to data.

**CHAPTER – 8**

**THE PROCESS**

Cancer prediction is a crucial application of machine learning in healthcare. With the advancement of technology, we can collect data using IoT sensors like wearable devices, smartwatches, and other IoT sensors. These sensors can measure vital signs like heart rate, body temperature, and blood pressure, which can be used to predict cancer. In this program, we will discuss the steps involved in cancer prediction using IoT sensors and machine learning.

Steps involved in cancer prediction using IoT sensors and machine learning:

**Collecting Data using IoT Sensors:**

The first step in cancer prediction is to collect the data. We can collect data using IoT sensors such as wearable devices, smartwatches, and other IoT sensors. These sensors can measure different parameters like heart rate, body temperature, blood pressure, and other vital signs.

**Sending Data to the Cloud:**

Once we have collected the data using IoT sensors, we need to send the data to the cloud for processing. We can use cloud services like AWS IoT or Microsoft Azure IoT to send the data to the cloud. The cloud provides a secure and scalable environment to process and store the data.

**Data Preprocessing:**

The data collected from the IoT sensors may have noise, missing values, or outliers. We need to preprocess the data to remove noise, handle missing values, and outliers. We can use data preprocessing techniques like data cleaning, data normalization, and data transformation.

**Feature Selection:**

Once we have preprocessed the data, we need to select the relevant features. We can use feature selection techniques like correlation analysis, mutual information, and chi-square test to select the relevant features.

**Training the Random Forest Algorithm:**

After selecting the relevant features, we can train the Random Forest algorithm. We can use libraries like scikit-learn in Python to train the Random Forest algorithm. We can split the data into training and testing sets and use the training set to train the Random Forest algorithm.

**Testing the Model:**

Once we have trained the Random Forest algorithm, we can use the testing set to test the model's accuracy. We can use evaluation metrics like accuracy, precision, recall, and F1-score to evaluate the model's performance.

**Deploying the Model:**

Once we have tested the model's accuracy, we can deploy the model. We can use cloud services like AWS Lambda or Azure Functions to deploy the model. These services provide a serverless environment to deploy the model.

**Predicting Cancer:**

Once the model is deployed, we can use it to predict cancer. We can send the new data collected from IoT sensors to the deployed model, and it will predict whether the person has cancer or not.

In conclusion, cancer prediction using IoT sensors and machine learning is a promising area in healthcare. We can collect data using IoT sensors, preprocess the data, select relevant features, train the model, and deploy it to predict cancer. With the help of cloud services, we can process and store a large amount of data securely and efficiently. This program is just a sample, and there are many variations possible based on the specific use case and data available. The potential applications of IoT sensors and machine learning in healthcare are vast, and we are yet to explore their full potential.

**PYTHON**

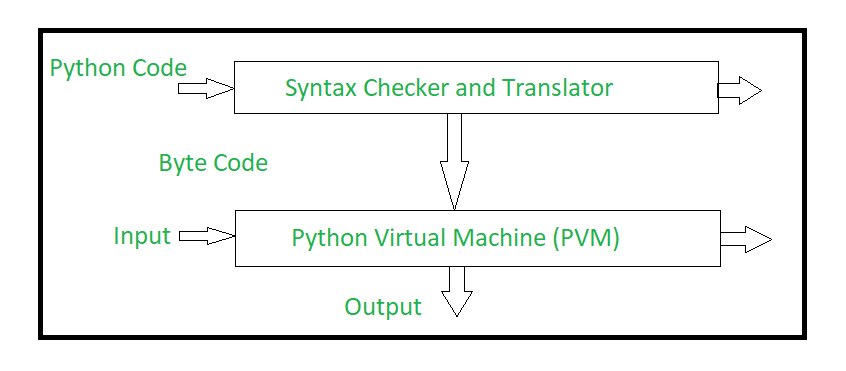
Python is a high-level, interpreted programming language that is widely used in various fields such as web development, data science, artificial intelligence, scientific computing, and automation. Python is known for its simplicity, readability, and ease of use, which makes it an ideal language for beginners as well as experienced programmers.



The key features and benefits of using Python:

* **Easy to learn and use:** Python has a simple and straightforward syntax that is easy to understand and read, making it ideal for beginners and experts alike.
* **Cross-platform:** Python can run on various platforms, including Windows, macOS, and Linux, making it a versatile language for developers.
* **Large standard library:** Python has a vast collection of built-in libraries and modules that make it easy to perform tasks such as file I/O, regular expressions, networking, and more.
* **Third-party packages:** Python has a vast ecosystem of third-party packages that can be easily installed using package managers such as pip, making it easy to extend the functionality of Python for specific tasks.
* **Data science and machine learning:** Python has become the go-to language for data science and machine learning due to its vast collection of data analysis libraries such as NumPy, Pandas, Matplotlib, and Scikit-Learn.
* **Web development:** Python can be used for web development using frameworks such as Django, Flask, and Pyramid.
* **Automation:** Python can be used for automating tasks such as web scraping, testing, and deployment using tools such as Selenium, Pytest, and Fabric.

Overall, Python is a versatile and powerful programming language that can be used for various tasks, from simple scripting to complex data analysis and machine learning. Its simplicity, readability, and vast collection of libraries and tools make it an excellent choice for developers and businesses alike.



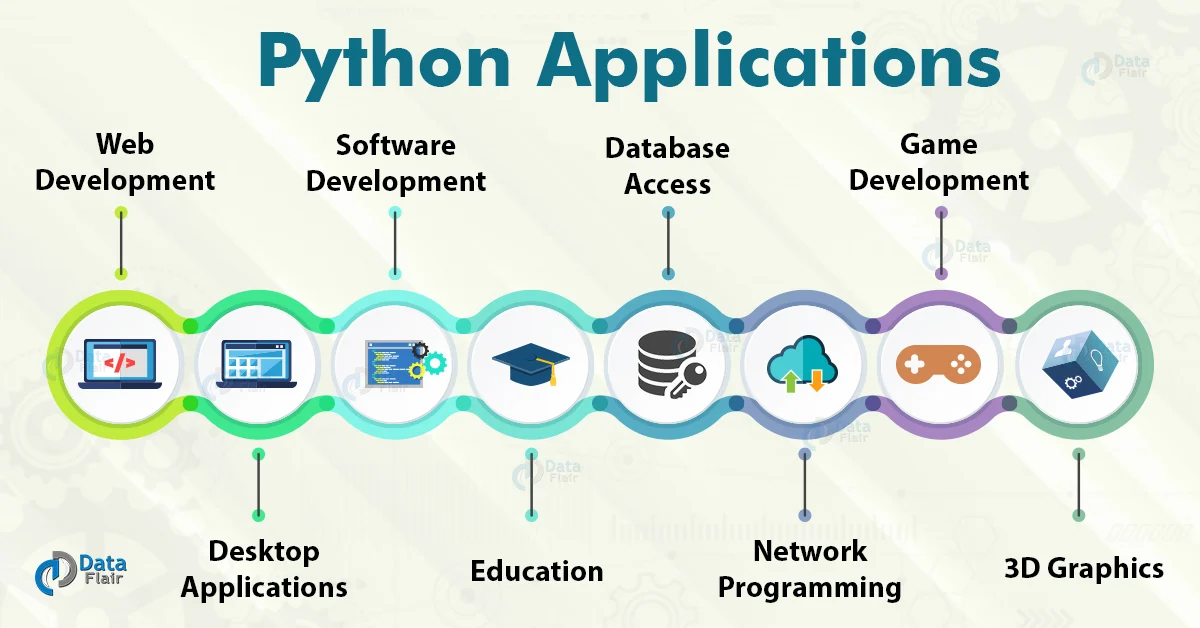
 Python uses code modules that are interchangeable instead of a single long list of instructions that was standard for functional programming languages. The standard implementation of python is called “cpython”. It is the default and widely used implementation of Python.

Python doesn’t convert its code into machine code, something that hardware can understand. It actually converts it into something called byte code. So within python, compilation happens, but it’s just not into a machine language. It is into byte code (.pyc or .pyo) and this byte code can’t be understood by the CPU. So we need an interpreter called the python virtual machine to execute the byte codes. 

The Python source code goes through the following to generate an executable code : 

* The python compiler reads a python source code or instruction. Then it verifies that the instruction is well-formatted, it checks the syntax of each line. If it encounters an error, it immediately halts the translation and shows an error message.
* If there is no error, i.e. if the python instruction or source code is well-formatted then the compiler translates it into its equivalent form in an intermediate language called “Byte code”.
* Byte code is then sent to the Python Virtual Machine(PVM) which is the python interpreter. PVM converts the python byte code into machine-executable code. If an error occurs during this interpretation then the conversion is halted with an error message.

**WHAT PYTHON TECHNOLOGY CAN DO :**

****

Python is a versatile programming language that can be used for various purposes, from simple scripting to complex data analysis and machine learning.

**Web development**: Python can be used for web development using frameworks such as Django, Flask, and Pyramid. These frameworks provide tools and libraries for building web applications, RESTful APIs, and microservices.

**Data analysis:** Python has a vast collection of data analysis libraries such as NumPy, Pandas, Matplotlib, and Scikit-Learn. These libraries allow developers to perform various tasks such as data cleaning, data manipulation, data visualization, and machine learning.

**Automation:** Python can be used for automating tasks such as web scraping, testing, and deployment using tools such as Selenium, Pytest, and Fabric. Python can also be used for creating bots for social media platforms, chatbots, and voice assistants.

**Game development**: Python can be used for game development using libraries such as Pygame and PyOpenGL. These libraries provide tools for creating 2D and 3D games, game engines, and game development frameworks.

**Desktop applications:** Python can be used for building desktop applications using frameworks such as Tkinter, PyQt, and PyGTK. These frameworks provide tools for creating graphical user interfaces, event handling, and interprocess communication.

**Network programming:** Python can be used for network programming using libraries such as socket, Twisted, and Pyro. These libraries allow developers to create network applications, servers, and clients.

**Artificial intelligence and machine learning:** Python has become the go-to language for artificial intelligence and machine learning due to its vast collection of data analysis libraries such as NumPy, Pandas, Matplotlib, and Scikit-Learn. Python can be used for creating neural networks, deep learning models, and reinforcement learning algorithms.

**Scientific computing:** Python can be used for scientific computing using libraries such as SciPy, NumPy, and Matplotlib. These libraries provide tools for scientific computing, numerical analysis, and data visualization.

Python is a versatile and powerful programming language that can be used for various tasks, from simple scripting to complex data analysis and machine learning. Its simplicity, readability, and vast collection of libraries and tools make it an excellent choice for developers and businesses alike.

**HOW JAVA TECHNOLOGY WILL CHANGE THE FUTURE**

Python is a versatile and powerful programming language that has been gaining popularity in recent years due to its simplicity, readability, and vast collection of libraries and tools. It has been widely adopted by developers, businesses, and academic institutions, and its use is expected to continue to grow in the future.

* **Increased efficiency and productivity:** Python’s simplicity and ease of use allow developers to write code more quickly and efficiently, leading to increased productivity. Its vast collection of libraries and tools also reduces the need for developers to write code from scratch, further improving efficiency.
* **Automation and artificial intelligence:** Python’s popularity in automation and artificial intelligence (AI) is expected to increase in the future. With the increasing demand for automation in various industries, Python can be used to develop robots, chatbots, and voice assistants. Python is also the language of choice for machine learning and data science, allowing developers to create intelligent applications that can learn and adapt to changing data.
* **Education and research:** Python’s simplicity and ease of use make it an ideal language for beginners and students. Python is widely used in academic institutions for teaching programming and data science courses. Python’s vast collection of libraries and tools also makes it an excellent choice for researchers in various fields, such as physics, biology, and social sciences.
* **Web development and cloud computing:** Python’s popularity in web development is expected to increase in the future. Python’s web frameworks, such as Django and Flask, allow developers to build web applications and microservices quickly and easily. Python is also widely used in cloud computing, with cloud providers such as Amazon Web Services and Google Cloud Platform offering Python support for building cloud applications.
* **Open-source and community-driven:** Python is an open-source language, which means that its source code is freely available to the public. This has led to a large and active community of developers who contribute to the language by creating libraries, tools, and frameworks. This community-driven approach has led to the creation of many useful resources, such as documentation, tutorials, and forums, making it easier for developers to learn and use Python.

**GOOGLE COLABORATORY :**

****

Colaboratory is a data analysis tool which combines code, output and descriptive text into one document

Colab provides GPU and is totally free. By using Google Colab, we can:

* Build your analytics products quickly in a standardized environments.
* Facilitates popular DL libraries on the go such as PyTorch, and TensorFlow
* Share code & results within your Google Drive
* Save copies and create playground modes for knowledge sharing
* Colab is runnable on the cloud or on [local server with Jupyter](https://research.google.com/colaboratory/local-runtimes.html)

Google Colaboratory, also known as Colab, is a free cloud-based platform that allows users to run and share Jupyter notebooks. Colab provides an easy-to-use and accessible environment for data analysis and machine learning tasks without the need for expensive hardware or software installations.

One of the key benefits of using Google Colaboratory is that it provides users with a ready-to-use computing environment. Colab comes preinstalled with Python, Jupyter notebook, and a range of popular Python libraries, such as NumPy, Pandas, and Matplotlib. This means that users can start working on their projects without having to spend time setting up a local environment.

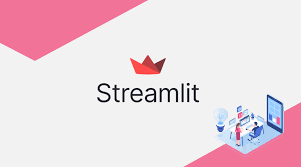
In addition to its ease of use, Google Colaboratory provides access to high-performance computing resources, such as GPUs and TPUs. These resources can be used to speed up computationally-intensive tasks, such as image and language processing, making Colab a great tool for researchers and data scientists.

Another advantage of using Google Colaboratory is its integration with other Google services, such as Google Drive and Google Sheets. Users can easily import data from these services into their notebooks, making it easier to work with and analyze data. Colab also allows users to collaborate on notebooks in real-time, making it an excellent tool for teams working on data analysis and machine learning projects.

Google Colaboratory is also a great platform for beginners learning Python programming, data analysis, and machine learning. The platform provides a beginner-friendly environment for students to learn and practice programming, and educators can create and share notebooks with their students.

In conclusion, Google Colaboratory is an excellent tool for data analysis and machine learning tasks. Its easy-to-use and accessible environment, access to high-performance computing resources, integration with other Google services, and beginner-friendly environment make it an ideal platform for researchers, data scientists, and students.

**STREAMLIT**

****

**Streamlit** is an open-source Python library that allows you to create web applications with interactive user interfaces for machine learning and data science projects. It’s designed to make it easy to build beautiful, customizable, and responsive web applications with minimal code. With Streamlit, you can create data-driven apps that can be easily shared and deployed to the cloud.

**Key features of Streamlit:**

1. **Easy-to-use:** Streamlit is very easy to use, even for beginners. You can create a web app with just a few lines of Python code.

**2. Fast prototyping:** Streamlit is designed to allow fast prototyping of data science projects. You can quickly test your ideas and iterate on them.

**3. Interactive widgets:** Streamlit provides a wide range of interactive widgets such as sliders, buttons, and text inputs that allow users to interact with your app.

**4. Built-in charts and graphs**: Streamlit has built-in support for creating charts and graphs with popular libraries like Matplotlib and Plotly.

**5. Customizable themes**: Streamlit allows you to customize the look and feel of your app with different themes and layouts.

**6. Deploy to cloud**: Streamlit makes it easy to deploy your app to cloud platforms like Heroku, AWS, and Google Cloud.

**Advantages of using Streamlit:**

**Easy to learn:** Streamlit has a simple and intuitive API, which makes it easy to learn and use.

**Rapid development:** Streamlit allows you to quickly build and test your ideas, so you can iterate faster and deliver projects more quickly.

**Data visualization:** Streamlit makes it easy to create interactive visualizations, which are essential for data science projects.

**Collaboration:** Streamlit is built for collaboration. You can easily share your app with others, and they can run it on their own machines.

**Deployment:** Streamlit makes it easy to deploy your app to the cloud, so others can access it from anywhere in the world.

**Functions of Streamlit:**

* **Text** : Streamlit provides several functions for displaying text, including `st.title()`, `st.header()`, `st.subheader()`, `st.text()`, and `st.markdown()`.
* **Widgets**: Streamlit provides a wide range of interactive widgets such as sliders, buttons, checkboxes, radio buttons, and dropdown menus. These widgets allow users to interact with your app and provide input.
* **Charts and graphs**: Streamlit has built-in support for creating charts and graphs with popular libraries like Matplotlib and Plotly. You can use functions like `st.line\_chart()`, `st.area\_chart()`, `st.bar\_chart()`, `st.pyplot()`, and `st.plotly\_chart()` to create these visualizations.
* **Dataframes**: Streamlit makes it easy to display Pandas dataframes with the `st.dataframe()` function.
* **Images:** Streamlit allows you to display images with the `st.image()` function.
* **File uploads:** Streamlit provides a function called `st.file\_uploader()` that allows users to upload files to your app.

**Conclusion**

Streamlit is an excellent tool for building data-driven web applications with Python. It’s easy to use, fast, and provides a wide range of interactive widgets, charts, and graphs. With Streamlit, you can quickly prototype your ideas and share your apps with others.

**CHAPTER – 7**

**RESULT & CONCLUSION**

* 1. **Conclusion**

The proposed cancer prediction system using IoT and cloud computing has several advantages over traditional cancer detection methods. One of the key benefits is its ability to provide real-time monitoring of patient data. With the use of sensors embedded in the human body, the system can continuously collect and analyze patient data, providing healthcare professionals with up-to-date information on the patient's condition.

Another advantage of the proposed system is its ability to securely store patient data in the cloud. This allows doctors and healthcare nurses to access patient information from anywhere, making it easier to monitor patients remotely and provide timely interventions. Moreover, the use of encryption ensures that patient data is protected from unauthorized access, ensuring the privacy and confidentiality of patient information.

The proposed system also provides patients with greater control over their medical data. Patients can access their medical records from anywhere, allowing them to take a more active role in their healthcare. This can lead to better patient outcomes, as patients are more likely to follow treatment plans when they are engaged in their care.

Overall, the proposed system has the potential to revolutionize the healthcare industry by improving cancer detection and treatment. It provides a more efficient and accurate method for detecting cancer, and has the potential to save lives by allowing for early detection and treatment. With further development and testing, the system could be implemented in healthcare systems worldwide, improving patient outcomes and reducing healthcare costs.

* 1. **Future Enhancements**

There are numerous future enhancements that can be made to the proposed cancer prediction system using IoT and cloud computing. By incorporating advanced AI and machine learning algorithms, expanding the system to cover more types of cancer, integrating with other healthcare systems, and developing user-friendly mobile applications, the system can be further improved to provide better healthcare outcomes for patients.

**REFERENCES**

[1]. Juhnyoung Lee, “A View Of Cloud Computing”, International Journal of Networked

and Distributed Computing, Vol. 1, No. 1 (January 2013), 2-8.

[2]. Rabi Prasad Padhy, Manas Ranjan Patra, Suresh Chandra Satapathy, “Design and

Implementation of a Cloud based Rural Healthcare Information System Model”,

UNIASCIT, Vol 2 (1), 2012, 149-157, ISSN 2250-0987.

[3]. S. Rashmi , Ghavghave, Deepali and M. Khatwar, “Architecture for Data Security In

Multicloud Using AES-256 Encryption Algorithm” , International Journal on Recent

and Innovation Trends in Computing and Communication (IJRITCC), ISSN: 2321-

8169,Volume 3, Issue 5, April 2017.

[4]. Uma Somani, Kanika Lakhani and Manish Mundra, “ Implementing Digital Signature

with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud

Computing”, International Conference on Parallel, Distributed and Grid Computing

(PDGC - 2010), pp. 211-216, Solan, India, Oct 2010.

[5]. B. Sri Varsha and, P.S. Suryateja, “Using Advanced Encryption Standard for Secure

and Scalable Sharing of Personal Health Records in Cloud”, International Journal of

Computer Science and Information Technologies(IJCSIT), ISSN 0975-9646, Volume

5, Issue 6 , 2014.

[6]. Aravindh Ramaswamy, Aswath Balasubramanian, Palaniappan Vijaykumar,

Varalakshmi P, “A Mobile Agent based Approach of ensuring Trustworthiness in the

Cloud”, IEEE-International Conference on Recent Trends in Information Technology,

ICRTIT 2011 978-1-4577-0590-8/11/$26.00 ©2011 IEEE, MIT, Anna University,

Chennai. June 3-5, 2011.

[7]. P.Varalakshmi, and Maheshwari.K., “Cost-Optimized Resource Provisioning in

Cloud”, 2013 International Conference on Recent Trends in Information Technology

(ICRTIT), ISBN:978-1-4799-1024-3/13/$31.00 ©2013 IEEE.

[8]. Emmanuel Effiok, Enjie Liu, Hong Qing Yu, and Jon Hitchcock, “A Prostate Cancer

Care Process Example of Using Data from Internet of Things”, 2015 IEEE

International Conference on Computer and Information Technology; Ubiquitous

Computing and Communications; Dependable, Autonomic and Secure Computing;

Pervasive Intelligence and Computing, DOI

10.1109/CIT/IUCC/DASC/PICOM.2015.340.

[9]. Sathish Kumar M and Balamurugan B, “A review on Performance Evaluation

Techniques in cloud”, 2017 Second International Conference on Recent Trends and

Challenges in Computational Models, DOI 10.1109/ICRTCCM.2017.29.

[10]. M.R.Aswin and M.Kavitha, “Cloud Intelligent Track – Risk Analysis And

Privacy Data Management In The Cloud Computing”, ISBN: 978-1-4673-1601-

9/12/$31.00 ©2012 IEEE.

[11]. Sridhar S, Venkatesh M, Sarveswaran V, Varalakshmi P, “Client Centric

Multitenancy for Cloud Services”, ISBN: 978-1-4673-1601-9/12/$31.00 ©2012 IEEE

239.

[12]. Ankush R. Deshmukh, Prof. R. V. Mante and Dr. P. N. Chatur, “Cloud Based

Deduplication and Self DataDestruction”, 2017 International Conference on Recent

Trends in Electrical, Electronics and Computing Technologies, DOI

10.1109/ICRTEECT.2017.26.

[13]. Ms. Shalini Ramanathan, Dr.Savita Goel and Mr.Subramanian Alagumalai,

“Comparison of Cloud Database: Amazon’s SimpleDB and Google’s Bigtable”, 2011

International Conference on Recent Trends in Information Systems, 978-1-4577-

0792-6/11/$26.00 ©2011 IEEE.

[14]. Nathezhtha.T and Vaidehi.V, “Cloud Insider Attack Detection Using Machine

Learning”, 978-1-5386-5853-6/18/$31.00\_c 2018 IEEE.

[15]. LuisCavo "Design of an area efficient crypto processor for 3GPP-LTE NBIoT

devices" https://doi.org/10.1016/j.micpro.2019.102899